East West University
Department of CSE

Project

<u>Project no : 2</u>

Submitted to: Dr. Taskeed Jabid
Chairperson, Associate Professor
Department of Computer Science & Engineering

Submitted by : Name: Sanjida Akter
ID: 2020-1-60-222

Name: Nazia Zaman
ID: 2020-2-60-016

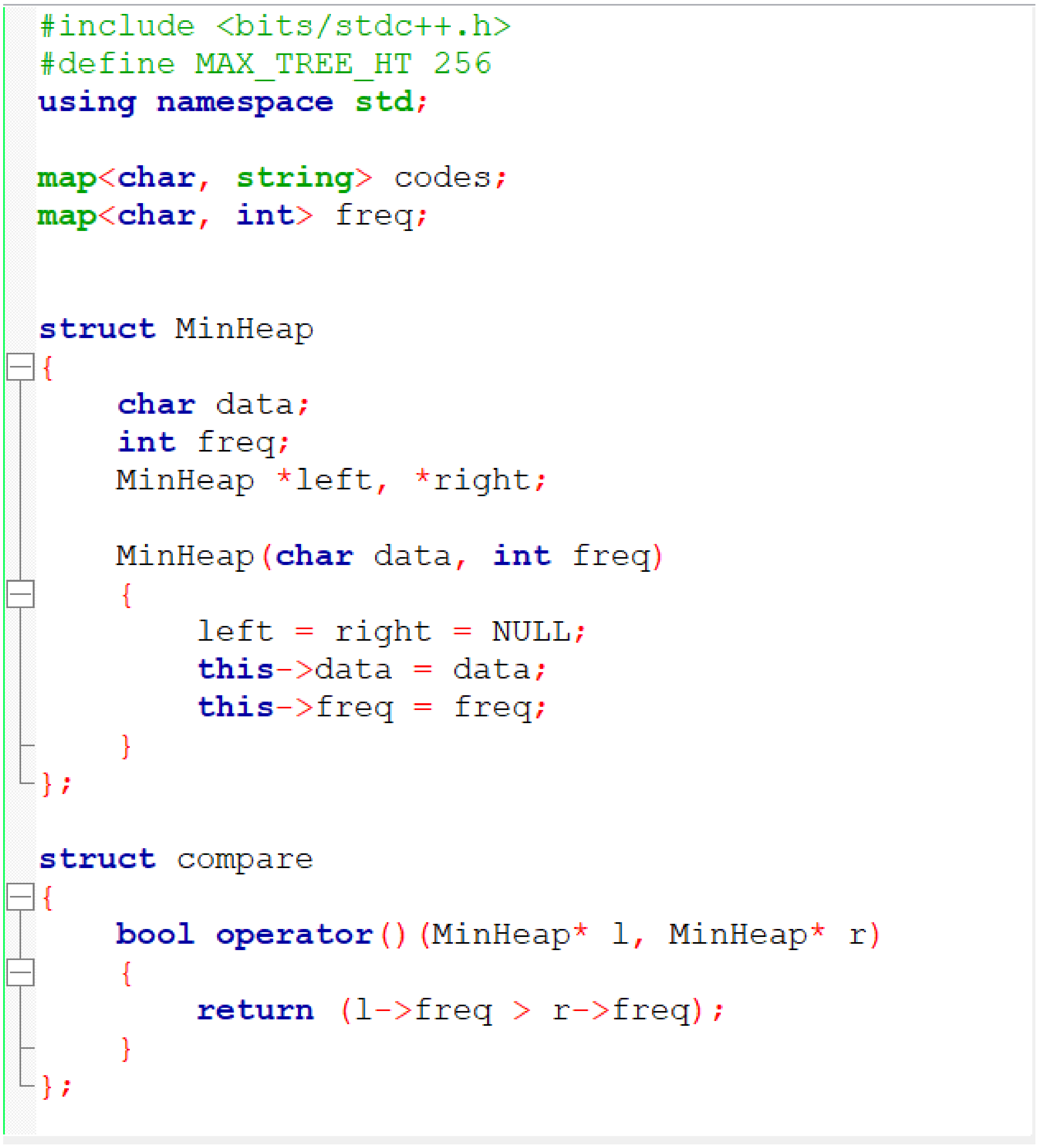
Name: Tarek Rahman
ID: 2020-1-60-218

Name: Ayesha Chowdhury
ID: 2020-1-60-003

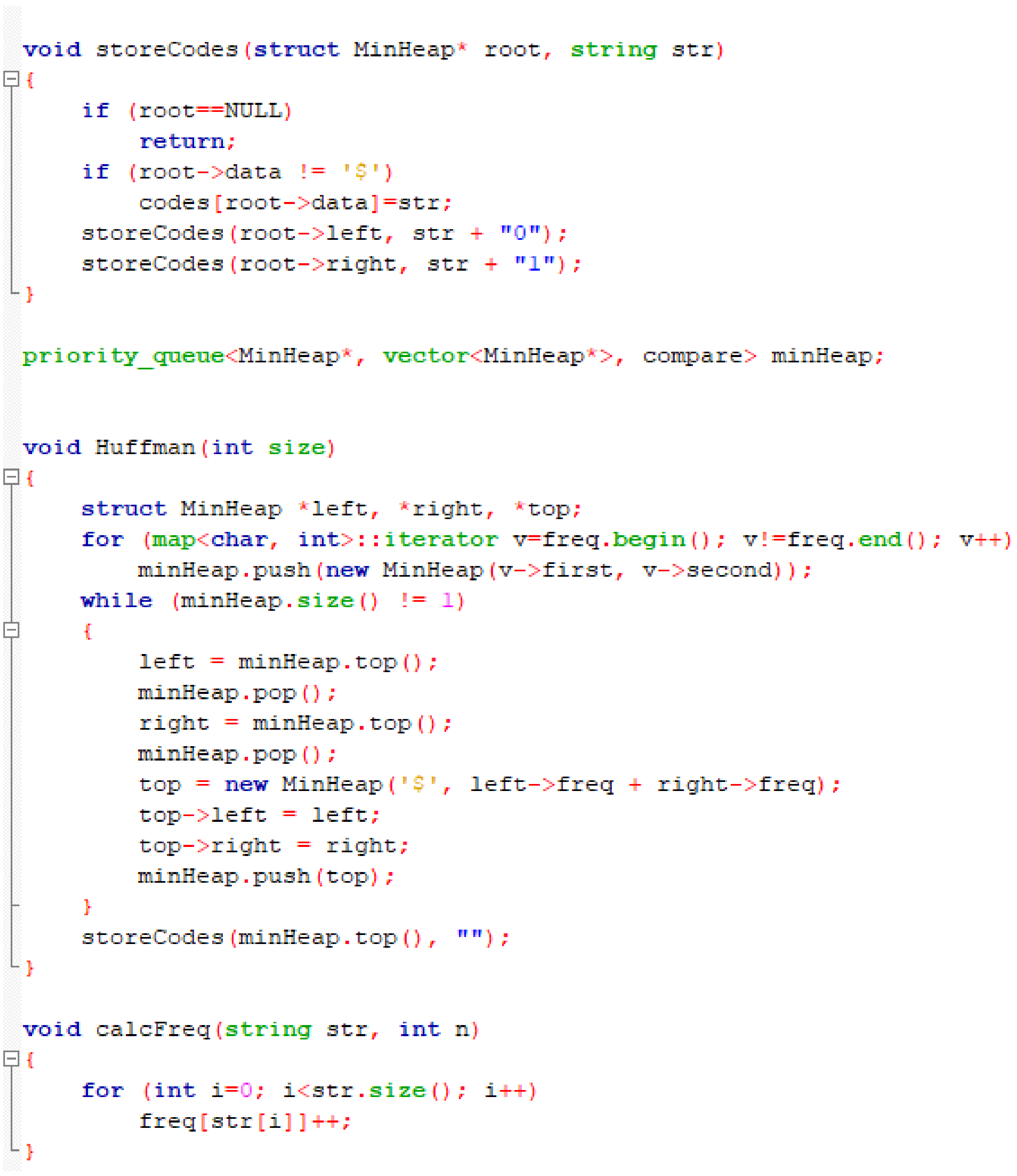
Name: Nadiatul Sharna
ID: 2020-1-60-023

Date of submission: 18/ 09/ 2022

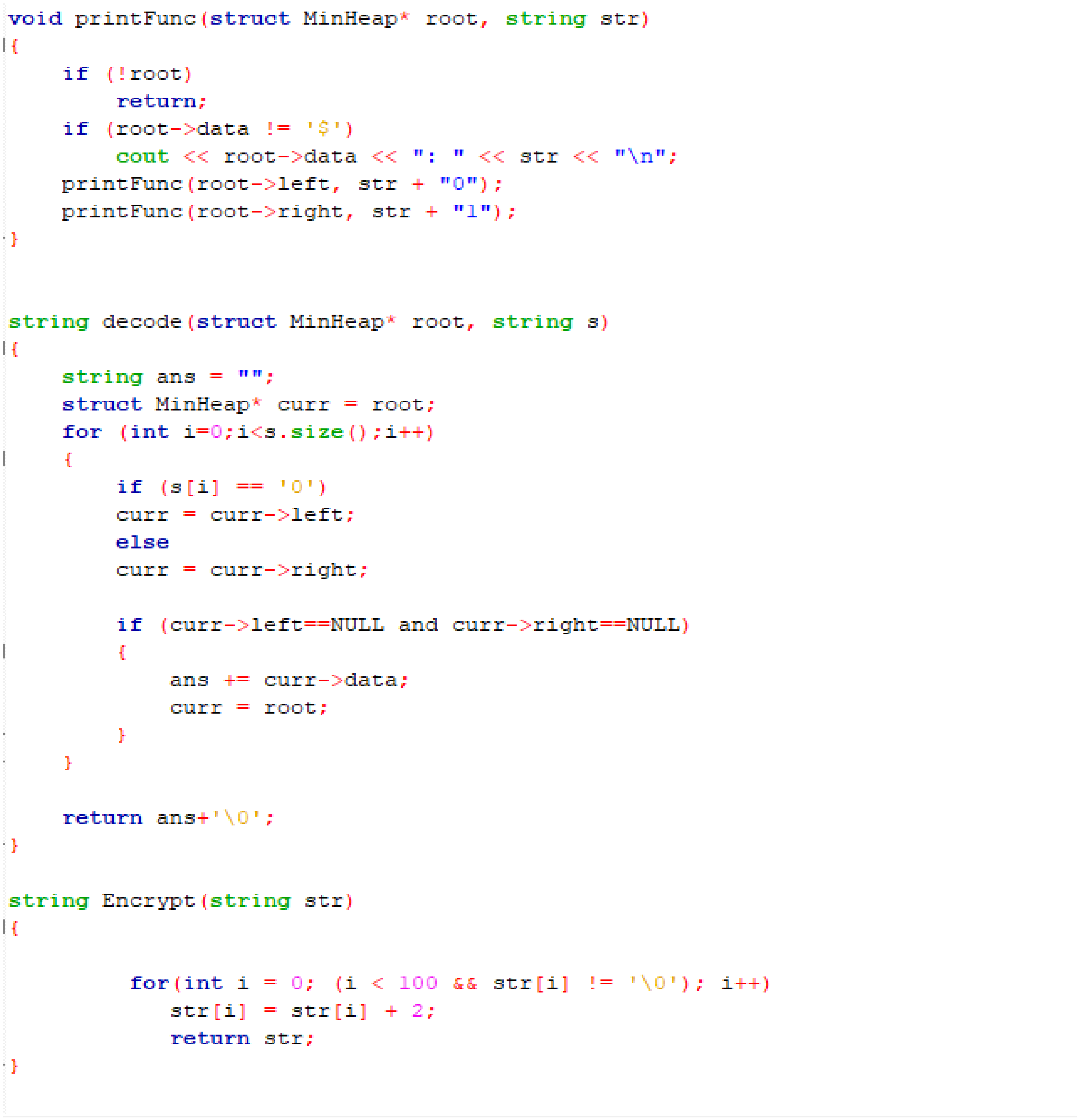
SOLUTION CODE:



```cpp
#include <bits/stdc++.h>
#define MAX_TREE_HT 256
using namespace std;

map<char, string> codes;
map<char, int> freq;


struct MinHeap
{
    char data;
    int freq;
    MinHeap *left, *right;

    MinHeap(char data, int freq)
    {
        left = right = NULL;
        this->data = data;
        this->freq = freq;
    }
};

struct compare
{
    bool operator()(MinHeap* l, MinHeap* r)
    {
        return (l->freq > r->freq);
    }
};
```



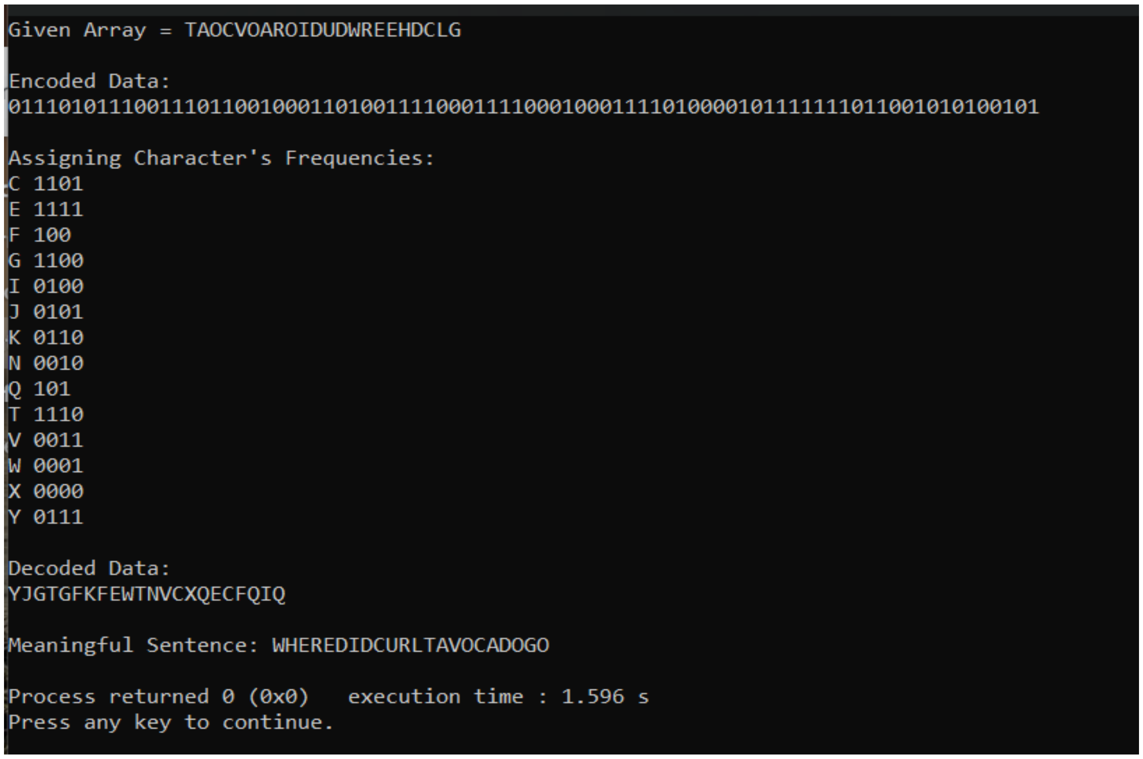
```cpp
void storeCodes(struct MinHeap* root, string str)
{
    if (root==NULL)
        return;
    if (root->data != '$')
        codes[root->data]=str;
    storeCodes(root->left, str + "0");
    storeCodes(root->right, str + "1");
}

priority_queue<MinHeap*, vector<MinHeap*>, compare> minHeap;


void Huffman(int size)
{
    struct MinHeap *left, *right, *top;
    for (map<char, int>::iterator v=freq.begin(); v!=freq.end(); v++)
        minHeap.push(new MinHeap(v->first, v->second));
    while (minHeap.size() != 1)
    {
        left = minHeap.top();
        minHeap.pop();
        right = minHeap.top();
        minHeap.pop();
        top = new MinHeap('$', left->freq + right->freq);
        top->left = left;
        top->right = right;
        minHeap.push(top);
    }
    storeCodes(minHeap.top(), "");
}

void calcFreq(string str, int n)
{
    for (int i=0; i<str.size(); i++)
        freq[str[i]]++;
}
```



```
void printFunc(struct MinHeap* root, string str)
{
    if (!root)
        return;
    if (root->data != '$')
        cout << root->data << ": " << str << "\n";
    printFunc(root->left, str + "0");
    printFunc(root->right, str + "1");
}


string decode(struct MinHeap* root, string s)
{
    string ans = "";
    struct MinHeap* curr = root;
    for (int i=0;i<s.size();i++)
    {
        if (s[i] == '0')
        curr = curr->left;
        else
        curr = curr->right;

        if (curr->left==NULL and curr->right==NULL)
        {
            ans += curr->data;
            curr = root;
        }
    }

    return ans+'\0';
}

string Encrypt(string str)
{

         for(int i = 0; (i < 100 && str[i] != '\0'); i++)
            str[i] = str[i] + 2;
            return str;
}
```

```
   string Decrypt(string str)
{
          for(int i = 0; (i < 100 && str[i] != '\0'); i++)
          str[i] = str[i] - 2;
       cout << "\nMeaningful Sentence: " << str << endl;
    return str;

}


int main()
{
    cout<<"Given Array = TAOCVOAROIDUDWREEHDCLG "<<"\n";
    string str = "WHEREDIDCURLTAVOCADOGO";
    string str1=Encrypt(str);

    string eString, dString;
    calcFreq(str1, str1.length());
    Huffman(str1.length());

    for (auto i: str1)
        eString+=codes[i];

    cout << "\nEncoded Data:\n" << eString << endl;

    cout << "\nAssigning Character's Frequencies:\n";
    for (auto v=codes.begin(); v!=codes.end(); v++)
        cout << v->first <<' '<< v->second << endl;

    dString = decode(minHeap.top(), eString);
    cout << "\nDecoded Data:\n" << dString << endl;

    string str2=Decrypt(str1);
    return 0;
}
```

Output:



```
Given Array = TAOCVOAROIDUDWREEHDCLG

Encoded Data:
0111010111001110110010001101001111000111100010001111010000101111111011001010100101

Assigning Character's Frequencies:
C 1101
E 1111
F 100
G 1100
I 0100
J 0101
K 0110
N 0010
Q 101
T 1110
V 0011
W 0001
X 0000
Y 0111

Decoded Data:
YJGTGFKFEWTNVCXQECFQIQ

Meaningful Sentence: WHEREDIDCURLTAVOCADOGO

Process returned 0 (0x0)   execution time : 1.596 s
Press any key to continue.
```

Discussion:

We have given a string which is "TAOCVOAROIDUDWREEHDCLG" and it is encrypted. This needs to decrypt. We need to make it a complete meaningful sentence. We are using Huffman coding for transmit the message.

Using the Huffman Coding technique, we can compress the string to a smaller size. Huffman coding first creates a tree using the frequencies of the character and then generates code for each character. Once the data is encoded, it has to be decoded. Decoding is done using the same tree.

Calculating the frequency of each character we need to sort them in a priority queue and will make a leaf node with each unique character. The minimum frequency will go to the leaf node the maximum will go to the right node. This is how we transmit data by this Huffman coding.

And this is the encoded code we are sending to our teammates. Then the teammates need to decrypt the code. And we have to traverse the tree which is made by the Huffman code and find the character. And we have to traverse from the root. And at the end they'll get a decoded string which is "YJGTGFKFEWTNVCXQECFQIQ". Then they have to use a decrypt function which will give them a meaningful sentence which is "WHEREDIDCURLTAVOCADOGO".